HOLLOWS: A Power-aware Task Scheduler for Energy Harvesting Sensor Nodes

JOAQUÍN RECAS PIORNO,[1,2,*] CARLO BERGONZINI,[3] DAVID ATIENZA[1] AND TAJANA SIMUNIC ROSING[3]

[1]*Embedded Systems Laboratory (ESL), Ecole Polytechnique Fédérale de Lausanne (EPFL), EPFL-STI-IEL-ESL 1015-Lausanne, Switzerland*

[2]*DACYA, Fac. de Informática, Complutense Univ. of Madrid (UCM) Prof. José García Santesmases sn, 28040 Madrid, Spain*

[3]*CSE Department, University of California San Diego (UCSD), La Jolla, CA 92093, USA*

ABSTRACT: Energy harvesting sensor nodes (EHSNs) have stringent low-energy consumption requirements, but they need to concurrently execute several types of tasks (processing, sensing, actuation, etc.). Furthermore, no accurate models exist to predict the energy harvesting income in order to adapt at run-time the executing set of prioritized tasks. In this article, we propose a novel power-aware task scheduler for EHSNs, namely, *HOLLOWS: Head-of-Line Low-Overhead Wide-priority Service*. HOLLOWS uses an energy-constrained prioritized queue model to describe the residence time of tasks entering the system and dynamically selects the set of tasks to execute, according to system accuracy requirements and expected energy. Moreover, HOLLOWS includes a new energy harvesting prediction algorithm, that is, *weather-conditioned moving average* (WCMA), which we have developed to estimate the solar panel energy income. We have tested HOLLOWS using the real-life working conditions of Shimmer, a sensor node for structural health monitoring. Our results indicate that HOLLOWS accurately predicts the energy available in Shimmer to guarantee a certain damage monitoring quality for long-term autonomous scenarios. Also, HOLLOWS is able to adjust the use of the incoming energy harvesting to achieve high accuracy for rapid event damage assessment (after earthquakes, fires, etc.).

Key Words: prioritized queue, power management, embedded systems.

INTRODUCTION

BATTERY-POWERED embedded sensor nodes (Motes (Crossbow, 2009); DuraNode (Park et al., 2005), etc.) are a very promising technology to achieve wide monitoring systems for environmental monitoring. These sensor nodes are able to gather data and periodically transmit it wirelessly to enable autonomous supervision of large natural environments and surfaces (e.g., natural parks, tunnels, bridges, etc.). However, in order to achieve a long-time operation with their limited battery sizes, they require very low power system solutions. As a result, a large percentage of these embedded sensor nodes tends to perform simple data collection (e.g., temperature or humidity sensing) and try to minimize the energy consumption due to communication, as this is the largest contributor since the complex filtering has been traditionally done in centralized hubs or intermediate sensor nodes (Raghunathan et al., 2002).

*Author to whom correspondence should be addressed.
E-mail: jrecas@fis.ucm.es

However, in the recent years, a very important progress towards ultra-low-power processing architectures has been achieved and last generation of embedded sensor nodes (Lin et al., 2005; Musiani et al., 2007) are able to perform much more intensive *in situ* filtering and signal processing computation. Thus, it is possible to enhance their flexibility to execute multiple concurrent algorithms and applications in the collected data from the environment in order to achieve complete natural environment evaluations and complex features extraction (e.g., average level of damage in bridges surfaces, pollution analysis in large surfaces of the forests, etc.).

These new processing capabilities and the need for advanced environmental signal processing in embedded sensor nodes have led to the development of a whole new set of applications and multi-tasking system designs that exploit these new node architectures, such as, NIMS (Pon et al., 2005) or Robomote (Sibley et al., 2002). Nonetheless, these systems require significant human maintenance, because their batteries need to be replaced regularly.

1045-389X/10/13 1317–19 $10.00/0 DOI: 10.1177/1045389X10377033

As a consequence, a very recent and important line of research in this area is the development of energy harvesting sensor nodes (EHSN). These new sensor nodes envisage the idea of self-maintained sensing systems by replacing traditional batteries with complex rechargeable energy storage systems, using combinations of batteries and supercapacitors, which are subsequently connected to different types of energy harvesting units (Raghunathan and Chou, 2006).

Many different types of energy harvesting technologies have been proposed nowadays, such as, solar, vibration, wind, piezoelectric, and thermoelectric. Among all these harvesting technologies, solar energy harvesting is currently showing the most promising energy versus area income trade-off (i.e., power-per-square centimeter), as shown in Raghunathan et al. (2005) or Musiani et al. (2007). However, even though solar panels can provide a significant amount of energy in specific natural working conditions (e.g., large intervals of sunny days), they are also subjected to very high variations of energy income due to variable weather conditions, which creates a large uncertainty on the expected quality and availability of the different EHSN. As a consequence, it is key to deploy new technologies and algorithms that can dynamically adapt the multi-tasking computation capabilities of each EHSN, according to the expected energy provided by the solar panel at each moment.

In this regard, we propose in this article a novel power-aware task scheduler for energy harvesting-based sensor nodes, namely, *HOLLOWS: Head-of-Line Low-Overhead Wide-priority Service*. This new scheduler for EHSNs includes an accurate and fast analytical solution based on prioritized queues for tasks with different priorities, which makes it capable to predict at run-time the mean waiting and residence time for different multi-level priority tasks in a wide range of EHSN working conditions. To this end, HOLLOWS only uses as input the expected execution time and inter-arrival time for each type of task that needs to be executed as well as the available energy in the node and an estimation of the expected energy harvesting income in the near future. As a consequence, we have also developed and incorporated in HOLLOWS a new energy prediction algorithm for solar panels of EHSN designs, that is, the weather-conditioned moving average (WCMA) predictor.

We have validated HOLLOWS using real-life working setups of Shimmer (Musiani et al., 2007), a last-generation structural health monitoring EHSN. Our results illustrate that our proposed scheduler solution for EHSN, which combines multi-priority queuing theory and weather-based energy prediction, maximizes the use of the energy income of solar panels (adapting the tasks to be executed to their variable energy incoming patterns), and reaching an estimation error of tasks waiting time of only 1%. Therefore, HOLLOWS is able to obtain 3× better predictions for variable-weather conditions than state-of-the-art energy prediction algorithms proposed for EHSN, such as, the exponentially-weighted moving average (EWMA) (Cox, 1961) energy prediction method. Overall, HOLLOWS is able to guarantee a regular level of accuracy in autonomous operation of EHSN setups for long periods of time, thus, maximizing the exploitation of energy harvesting devices. Furthermore, HOLLOWS is highly versatile and can react to very critical and unexpected natural events (e.g., fires, earthquakes, etc.), such that it can dynamically change (in few seconds) its operation requirements in order to use the expected energy harvesting income to provide the highest possible instantaneous sensing accuracy in a desired time interval.

The rest of the article is organized as follows. First, the section 'Related work' provides an overview of the related work on energy prediction algorithms, EHSN architectures and energy harvesting methods. Next, the section 'Architectures of Energy Harvesting Sensor Nodes' summarizes the main features of EHSN platforms. Then, 'Task Scheduling for Energy Harvesting Sensor Nodes' section introduces our proposed HOLLOWS task scheduler for EHSN. Next, the section 'Experimental Setups of EHSNs and Results' presents our experimental results for the real-life Shimmer platform. Finally, in the last section, we draw the main conclusions of this work.

RELATED WORK

There are many different types of EHSN architectures. While all these architectures are designed to be self-powered and must imply very limited maintenance, their target application (environmental monitoring, building surveillance, water pollution analysis, etc.) implies different variations of signal processing applications and sensors to be used. Representative examples of EHSN of their different scopes are Heliomote (Lin et al., 2005), which is used for light, humidity, and environment temperature monitoring, the video surveillance sensor node presented by Magno et al. (2008), and Shimmer (Musiani et al., 2007). All these nodes rely on solar panels to harvest energy from the environment for normal operation. Furthermore, other EHSN platform can use multiple energy sources as AmbiMax (Park and Chou, 2006), which is able to harvest solar wind, thermal, and vibrational energy.

In all the previous architectures, a major role is played by the existing energy harvesting technologies. Nowadays, extensive research has been performed in this area. Thus, different types of energy harvesting technologies have been proposed, such as, solar panels (Raghunathan et al., 2005), vibrational systems (Meninger et al., 2001), wind shells (Morais et al.,

2008), and thermoelectric mechanisms (Mateu et al., 2007). Nonetheless, all these energy harvesting technologies are unpredictable in many cases, which implies that it is not possible to guarantee a maintained energy level for EHSN solutions without extensive power management and energy harvesting prediction studies (Raghunathan and Chou, 2006; Morais et al., 2008). In this regard, different estimation techniques have been developed for solar energy, which is the most predictable one, and can be modeled to predict the expected availability at a given time within some error margin (Kansal et al., 2007). Also, Esram and Chapman (2007) propose the use of maximum power point tracking (MPPT) techniques to effectively monitored and harvest the maximum amount of energy possible by varying the orientation of the solar panel. Hence, solar energy harvesting has proven to achieve so far the most promising energy versus area income trade-offs (i.e., power-per-square centimeter), as proven by Raghunathan et al. (2005) or Musiani et al. (2007). In any case, even though solar panels can provide a significant amount of energy in specific natural working conditions (e.g., large intervals of sunny days), they are also exposed to large variations of energy income due to variable weather conditions, which creates a large uncertainty on the expected quality and availability of the different EHSN. As a consequence, it is key to deploy new technologies and algorithms specifically targeting the features (e.g., multi-tasking computation complexity for EHSN, quality of service for different EHSN working environments, etc.).

Due to the variable income of energy through energy harvesting technologies, power and energy management is a major research topic in EHSN at present. In particular, research in this area targets maximum lifetime of the sensor nodes, while meeting the performance demands of their high-level applications as well as their wired or wireless connectivity (Marbach, 2007). In fact, different power management techniques have been studied to achieve trade-offs between performance and power consumption, namely, as duty-cycle techniques (Kansal et al., 2004), dynamic voltage and frequency scaling (Yuan and Qu, 2007; Raghunathan et al., 2001), task migration (Rong and Pedram, 2003), or low-power operation modes (Benini et al., 2000). Also, in the last few years, research starts being performed to provide power management in EHSN platforms that must include prioritization, as these new platforms must execute multiple sensing and processing tasks with different priorities each. In fact, Raghunathan et al. (2001) and Moser et al. (2006) have shown the benefits of using adaptive scheduling methods including priorities for power management. Nonetheless, these systems need to precharacterize the different working conditions of multiple sets of tasks, which implies extensive experimental validation before the power management strategy can be tuned correctly, which is a very strong limitation. In contrast, we propose in this article an adaptive, stochastic, and multi-queuing task scheduler, which is able to define the prioritization between tasks of multiple priorities just by using a very limited design time information, that is, the knowledge of the independent task features (i.e., power and deadline requirements), and the defined priority between them by the EHSN designer.

In addition, stochastic models for wireless sensor networks has been used in the literary to try to model the wireless communication channel. In particular, Marbach (2007) proposes a distributed scheduling and active queue management mechanism for wireless networks. This approach is based on a random access scheduler where the transmission attempt probability depends on the local backlog, and the performance of the resulting protocol is modeled as an optimization problem, which tries to maximize throughput and fair bandwidth allocation. Similarly, Shuman and Liu (2006) consider the problem of conserving energy in a single node in a wireless sensor node by turning off the radio for fixed periods of time. This approach tries to achieve sleep control management using a mixed cost function for both energy consumption and performance penalty due to retransmission costs for backlogged packets. Another idea in this area is to exploit the theory of continuous-time Markovian decision processes, as done by Rong and Pedram (2003). In this case, it is proposed the use of a stochastic model for dynamic power management, which dynamically monitors the channel conditions and the server behavior, and then defines a client-side power management policy. However, in all these previous task schedulers with power management considerations, the prioritization concept is used only at the level of modeling the wireless channels, which requires significant overhead due to the propagation of the status of the sensor network, and it is not considering the internal status of the node itself. In this article, HOLLOWS includes a stochastic model not to describe the wireless channel, but to describe the internal behavior of a single node with multiple levels of priority between the types of tasks that need to be executed, and adjust which ones should be executed according to the desired level of accuracy and the predicted level of energy harvesting income.

Regarding energy management with performance constraints in EHSN platforms, several energy prediction algorithms have been developed up today. These algorithms try to perform energy management by scheduling the workload not only taking into account the actual level of energy stored in the node, but also the incoming energy. On the one hand, off-line solar prediction algorithms based on mean expected values have been recently proposed. In Suehrcke and McCormick (1992), it is shown that the average-daily solar system

performance may be calculated from the product of clear-sky solar performance and the average time fraction of clear sky. This approximation greatly simplifies the solar system performance prediction, but it does not offer specific energy guarantees at certain daily intervals, and is not suitable for short-term predictions. Also, Iqdour and Zeroual (2007) introduce a new method for modeling daily sun radiations, based on Takagi–Sugeno fuzzy systems. This method uses a non-linear technique, defined by a set of If–Then rules with linear consequent parts, which establish a local linear input–output relationship between the variables of the model. Then, the parameters of the model are identified using the fuzzy clustering combined to the least square algorithm. While this model produces accurate results, it demands very high computation and power figures, making this algorithm not applicable to EHSN systems, as we target in this work.

Finally, a very well-established and low-cost (in terms of computation needs) energy prediction algorithm, suitable for run-time energy harvesting prediction, is the EWMA algorithm (Cox, 1961; Hunter, 1986; Vigorito et al., 2007; Kansal et al., 2007). The method has been designed to exploit the diurnal cycle in solar energy and to adapt to seasonal variations. To this end, EWMA calculates the value of energy likely to be harvested at a particular time as a weighted average of the energy received at the same time over a set of previous days (Cox, 1961). Although EWMA-based algorithms for EHSN systems are accurate for consistent weather conditions, when cloudy and sunny days are mixed, the high prediction weight given to the recent days energy values introduces significant prediction errors (see 'Experimental Setups of EHSNs and Results' section for more details). Therefore, in order to prevent this problem, we introduce in our task scheduler (HOLLOWS), a new prediction algorithm, that is, the WCMA energy harvesting predictor, which not only takes into account the solar conditions at a certain time of the day, but it also suitably self-adjusts the energy intake estimation for fast changing weather conditions during a single day.

ARCHITECTURES OF ENERGY HARVESTING SENSOR NODES

Current EHSN architectures need to execute multiple tasks, which include sensing multiple materials and environmental reactions, executing complex signal processing algorithms within the sampled data (e.g., filtering, feature extraction, data compression, path planning), wireless communications, solar panel energy harvesting monitoring, and actuation in many cases. Therefore, the common set of components of all these EHSN architectures are a low-power digital signal processing (DSP) microcontroller (Instruments, 2009), which applies a certain energy management algorithm and manages the energy harvesting components (e.g., a solar panel), an energy storage unit (e.g., batteries or supercapacitors; Simjee and Chou, 2006) and different types of actuation devices, sensors, processing applications, and radio communication devices. Figure 1 shows a general architecture of a typical EHSN. Various representative examples of such an EHSN architecture are Robomote (Sibley et al., 2002), Everlast (Simjee and Chou, 2006), DuraNode (Park et al., 2005), Medusa MK-2 (Savvides and Srivastava, 2002), and SHiMmer (Musiani et al., 2007).

In particular, in the following paragraphs we describe the Shimmer platform (Musiani et al., 2007), which is an active sensing platform for structural health monitoring or material damage identification, that will be used during the rest of this work as real-life EHSN case study in all the experimental results. The main components and characteristics of Shimmer are:

- Wireless radio link: it uses a Maxstream XBee module (Maxstream, 2009).
- High complexity computing capabilities: it includes a TI TMS320C2811 DSP microcontroller (Instruments, 2009).
- Actuating and sensing capabilities: it uses a matrix of 16 lead-zirconate-titanate piezoelectric transducers (PZTs).
- Energy harvesting device: it includes a PowerFilm solar panel (PowerFilm, 2009).
- Energy storage unit: it can use a supercapacitor, a rechargeable battery or a combination of both.

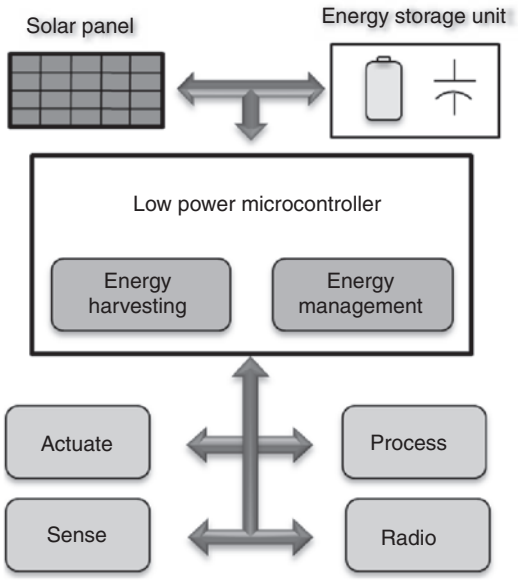

Figure 1. *Overview of representative EHSN architectures.*

– Multitask capabilities: it incorporates a real-time operative system, that is, FreeRTOS (FreeRTOS, 2009).

Structural damage can be defined as any geometric or material change introduced into a system that negatively impacts its current or future performance (Farrar and Worden, 2007). Thus, the PZTs included in Shimmer are used to generate sensing signals that perform the role of both an actuator and a sensor. Indeed, since the electrical impedance of a PZT is directly related to the structure's mechanical impedance, this impedance-based method uses high-frequency vibrations to monitor the structure's mechanical impedance in order to detect and locate damage in a structure (Park et al., 2003).

Then, structural health monitoring can be separated into two basic categories, namely, periodic lifetime monitoring and rapid event assessment. While periodic lifetime monitoring seeks to identify damage that accumulates over a long period of time, rapid event assessment addresses the need to obtain data from a structure immediately following a significant event, such as an earthquake, a fire, an accident, etc. In both types of categories of structural health monitoring applications, it is required the knowledge of the undamaged state of the structure and a continuous comparison of periodic measurements. To this end, data is first collected from the system using sensors. Then, normalization and cleansing of the data is performed to distinguish the damage due to external and environmental factors. Next, feature selection and information condensation is performed to reduce the overall amount of structural data to be handled. A feature is a specific property of the monitored system that can be used to identify damage and can be extracted from the sensed data. Finally, statistical model development for feature discrimination is applied, which involves the evaluation of the found features to quantify the damage. In particular, statistical modeling can refer to three types of algorithms: group classification, regression analysis, and outlier detection.

All those tasks are very computational intensive and require a significant amount of energy in EHSN platforms. Morever, it is not possible to globally optimize their multi-task execution pattern, as these tasks can be executed using many different configurations to achieve different accuracy levels in the structural health monitoring process. For instance, using three types of tasks, that is, actuation/sample (A), processing (P), and transmission (T), some of the possible real-life task execution patterns in Shimmer are:

– Actuate/Sample-Process-Send: This is the most general (and energy demanding) pattern, which sequentially performs: $A\ P\ T\ A\ P\ T\ A\ P\ T\ \ldots$
– Actuate/Sample-Send: If the consumption of the radio is lower than the processing, then the pattern: $A\ T\ A\ T\ A\ T\ A\ T\ldots$ may be used. This pattern can be very useful when the Shimmer node must provide real-time data streaming, but a significant part of energy is consumed in the wireless communication.
– Multiple Actuate/Sample: The pattern: $A\ A\ A\ A\ P\ T\ldots$ can be used to actuate multiple times before processing. This pattern is relevant because it enables considering multiple sets of samples together before transmitting an overall damage result (i.e., the average damage value from multiple PZTs).
– Event Triggering Sending: In some systems, the wireless communication may not always be available (e.g., the radio is momentarily in sleep mode). Hence, the pattern: $A\ P\ A\ P\ A\ P$ can be applied while the radio operation is resumed, which is when the transmit task (T) will be applied.

In the aforementioned EHSN architecture of the Shimmer node, where multiple tasks need to be executed (e.g., actuate, sense, process, radio sending, etc.) and multiple possible task-execution patterns can be applied in different working conditions, even the most effective energy harvesting mechanisms nowadays (i.e., solar panels) are not able to provide enough energy to execute all the necessary tasks to achieve always a 100% damage estimation precision. Hence, the definition of an appropriate power-aware task scheduler is a must. This scheduler needs to deploy suitable energy management algorithms, targeting the working conditions of EHSN, and must efficiently use both the stored energy and the energy entering the system through the included energy harvesting device.

TASK SCHEDULING FOR ENERGY HARVESTING SENSOR NODES

To be able to suitably define the pattern of tasks to be executed in EHSN platforms, according to the demanded precision at each moment in time and the available energy in the system, the task scheduler of each EHSN needs to incorporate an accurate system behavior model for a multi-task execution environment. Thus, in this section we present HOLLOWS, a power-aware task scheduler targeting specifically EHSN working environments. This new scheduler provides an efficient way to dynamically explore the trade-offs between the accuracy of the measurements of the sensor node and its consumed energy, based on the tasks characteristics. HOLLOWS has the structure shown in Figure 2. As this figure shows, HOLLOWS includes two fundamental parts. First, it consists of an integral EHSN behavior model, based on prioritized

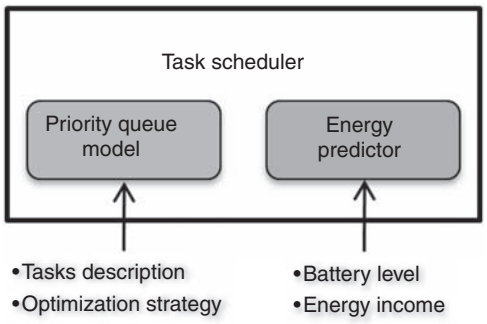

Figure 2. *Overview of the proposed HOLLOWS power-aware scheduler for EHSN.*

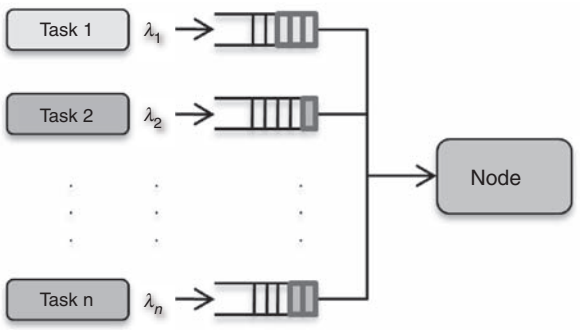

Figure 3. *Multi-task EHSN execution model.*

queues for multiple types of tasks. Second, it includes a new energy income prediction algorithm for EHSNs.

As shown in Figure 2, HOLLOWS requires to operate the following inputs of optimization strategies and specifications of the tasks to be executed in each EHSN:

- Specification of execution time and energy consumption: every task has to be characterized by a typical execution time and energy consumption value. Furthermore, HOLLOWS can also operate with multiple values and variations over the mean (for energy and execution time).
- Priority of each task: all the tasks in the system must have an associated priority, otherwise the minimum priority is automatically assigned to that task.
- Definition of the number of queues in the system or grouping strategy of similar tasks with the same priorities: the EHSN designer must indicate general rules that define in which queue of the node a new task, which arrives into the system, must be placed.
- Selection of optimization strategy: an optimization metric and policy must be defined by the node designer. This strategy can be as simple as specifying the fixed deadline of the tasks that will be executed or the desired residence (or waiting) time of the tasks in each queue, or as complex as a multi-objective optimization strategy like the guaranteed amount of energy that the system must kept in its internal storage unit in case of an emergency to be served with a certain accuracy level. Furthermore, HOLLOWS allows to modify at run-time the optimization strategy and the tasks characteristics, according to the dynamic user's requirements.

Using the previous system characterization, HOLLOWS uses an energy-constrained prioritized queue model to compute the residence time of the different tasks that enter the system. In addition, to analyze the available energy in the system in the near future, it also includes a new energy harvesting prediction algorithm, that is, the WCMA predictor.

In the following sections we detail the different components of HOLLOWS, namely, the head-of-line priority service with energy constraints that estimates the task execution delay and the modeled multi-priority system execution model for EHSNs, and the proposed WCMA energy harvesting predictor.

Delay in Energy Harvesting Systems

As the microcontrollers used in EHSN architectures are capable of executing concurrently multiple tasks with different priorities, in order to maximize the performance of the node under tight performance constraints, a priority scheduling system needs to be applied to model the overall node behavior and to give more resources to specific tasks. Thus, we first analytically describe the mean residence time for EHSN systems, with the priority service discipline presented in Figure 3. This figure shows a general task (T_k) that has a higher priority than T_{k+1}, that is, T_1 and T_n are the highest and lowest priority tasks, respectively.

HOLLOWS uses separate queues for different priority tasks and implements a non-preemptive tasks execution discipline. Thus, when the node becomes available, it selects a new task to be executed from the highest non-empty queue. This discipline is referred as *head-of-line priority service* (Leon-Garcia, 2008).

In systems without energy constraints, when the node becomes available, the head-of-line priority service mode guarantees that it is always possible to execute a new task. However, in energy harvesting nodes with a finite energy storage unit, even if the node is free to execute a task, it is possible that not enough energy is available to do so. Thus, the energy availability value must be taken into account when studying the residence time for the different tasks that need to be executed into the node.

As a result, HOLLOWS extends the head-of-line priority service basic discipline to include the EHSNs energy constraints in the computation of the residence time. Consequently, we first define $E[E_k]$ as the total estimated energy needed to execute a new task T_k that enters the system. Then, this amount of energy is

calculated as the sum of the needed energy to execute all the tasks of higher or equal priority that are waiting in the system ($E[J_k^W]$), all the higher priority tasks that arrive during the waiting time ($E[J_k^A]$), the energy needed to execute the task ($E[J_k]$), and the residual energy needed to finish the task currently in service ($E[R_J]$):

$$E[E_k] = E\left[J_k^W\right] + E\left[J_k^A\right] + E[J_k] + E\left[R^J\right] \quad (1)$$

Therefore, using this new equation of the total energy consumption for each task, we can introduce a novel priority-queue model (cf. section 'Delay in Priority Queue Systems') that provides us the necessary information to compute this total energy. Thus, given the total energy needed to execute a new task ($E[E_k]$), HOLLOWS can predict if a new task that arrives into the system can be executed when its turn comes or if it must wait until enough energy is harvested.

DELAY IN PRIORITY QUEUE SYSTEMS

Using the Shimmer node as case study, we have characterized the arrival patterns of different tasks to be executed in EHSN systems. To this end, we have used the high performance wireless research and education network or HPWREN Network (HPWREN, 2009), where the Shimmer node has been deployed (Musiani et al., 2007). The HPWREN network aggregates the results of several sensor nodes deployed in the field to gathers data from multiple sensing sources and forwards this data to the HPWREN backbone. Several node traces obtained from HPWREN have been studied to characterize the task inter-arrival time of different Shimmer nodes. Figure 4 represents the inter-arrival time of one sensor node.[1] Thus, Figure 5 illustrates how we have performed an exponential fit of the experimental data of the tasks arrival trace, obtaining a mean of $1/\lambda = 1.358$ ms with a 95% confidence interval of the parameter estimate of 1.343 ms and 1.373 ms, which is very close to the mean value.

As a result, the overall execution system in Shimmer nodes can be modeled as an extended version of the general $M/G/1$ queue (Leon-Garcia, 2008). This is a single-server queue where the tasks arrive according to a Poisson process of rate λ, and so the inter-arrival times follow an independent identically distributed exponential random variable with mean $1/\lambda_k$. Then, the service time follows a general independent identically distributed random variable distribution and it has an expected execution time $E[\tau_k]$. In addition, the inter-arrival and service times are independent and the queues can

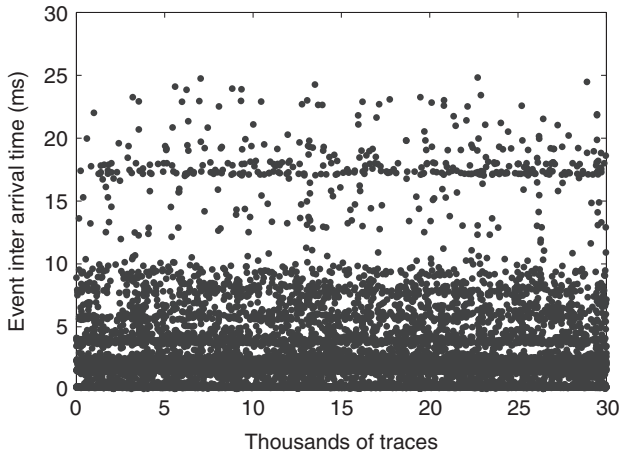

Figure 4. *HPWREN node inter-arrival requests.*

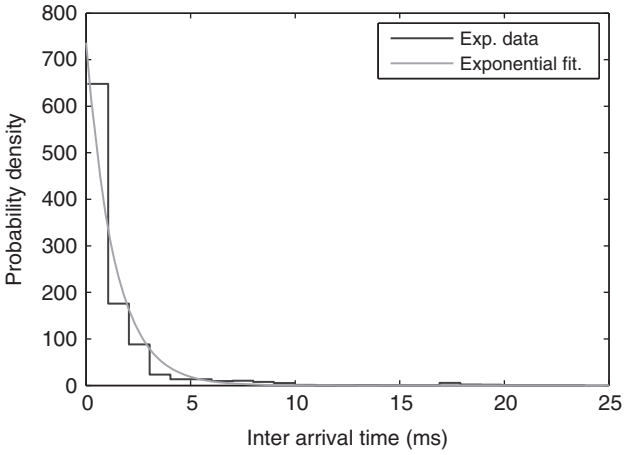

Figure 5. *HPWREN node requests trace exponential fit.*

accommodate an infinite number of elements. The model uses separate queues for different priority tasks and uses non-preemptive schedule. As a result, we define the server utilization of a task T_k as:

$$\mu_k = \lambda_k E[\tau_k]. \quad (2)$$

The estimated waiting time in a queue for the most priority task (T_1) can be computed as (see Leon-Garcia (2008) for more details):

$$E[W_1] = E[R^T] + E[N_{q1}]E[\tau_1] = \frac{E[R^T]}{1-\mu_1}, \quad (3)$$

where $E[R^T]$ is the residual service time of a task found in service, and $E[N_{q1}]$ is the expected number of tasks of type 1 found in the queue. T_2 has less priority than T_1; thus, to calculate the computation of the queue waiting

[1]The original tasks arrival request trace can be consulted on-line at HPWREN (2009).

time for T_2, we take into account the number of elements in the T_1 queue and the expected T_1 arrivals during the waiting time. Therefore, the queue waiting time can be defined as follows:

$$E[W_2] = E[R^T] + E[N_{q1}]E[\tau_1] + E[N_{q2}]E[\tau_2] + E[M_1]E[\tau_1], \quad (4)$$

where $E[N_{q2}]$ is the expected number of T_2 tasks found in the queue and $E[M_1]$ is the number of expected arrivals of T_1 during the waiting time. Hence, it can be proven (Leon-Garcia, 2008) that the queue waiting time is equal to:

$$E[W_2] = \frac{E[R^T]}{(1-\mu_1)(1-\mu_1-\mu_2)}. \quad (5)$$

Furthermore, in general, for a task T_k, it is equal to:

$$E[W_k] = \frac{E[R^T]}{(1-\mu_1-\cdots-\mu_{k-1})(1-\mu_1-\cdots-\mu_k)} \quad (6)$$

The task found in service by an arriving task can be of any type, so $E[R^T]$ is the residual service time of tasks of all types:

$$E[R^T] = 1/2\sum_{i=1}^{n}\lambda_i E[\tau_i^2], \quad (7)$$

where $E[\tau_i^2]$ is the second moment of the service time of T_i.

Using this model, provided the inter-arrival time of the different tasks ($1/\lambda_k$), the expected execution time ($E[\tau_k]$), and the second moment of the expected execution time ($E[\tau_i^2]$), it is possible to compute the total expected waiting time, that is, the time spent in a queue plus the execution time, as follows:

$$E[T_k] = E[W_k] + E[\tau_k]. \quad (8)$$

In order to illustrate how this queuing model is used in HOLLOWS for EHSN systems, Figure 6 shows a real-life working setup of the Shimmer node including three types of tasks: *Sense*, *Process* and *Transmit*, with the *Sense* and *Transmit*, as the highest and lowest priority task, respectively.

If we consider that the execution time follows an exponential random variable, according to the inter-arrival request pattern characterized in the section 'Delay in Energy Harvesting Systems', the system can be described as an $M/M/1$ queuing model. In this case, the second moment of the execution time, as proven by Leon-Garcia (2008), is $E[\tau_k^2] = 2 \times E[\tau_k]^2$. Hence, the residual service time using the Table 1 values for the tasks is:

$$E[R^T] = \sum_i \lambda_i E[\tau_i]^2 = 182.3\,\text{ms}. \quad (9)$$

Next, using Equation (6) we can compute in HOLLOWS the estimated queuing time for a new *Transmit* task that arrives into the EHSN system as:

$$E[W_T] = \frac{E[R^T]}{(1-\mu_S-\mu_P)(1-\mu_S-\mu_P-\mu_T)} = 444.3\,\text{ms}. \quad (10)$$

Finally, the mean residence time in the system for a new *Transmit* task is the queuing waiting time plus the execution time, which is 491.4 ms. A similar study can be done for the *Sense* and *Process* tasks, obtaining an estimated waiting time of $E[W_S] = 182.3$ ms and $E[W_P] = 283.6$ ms, and residence time of $E[T_S] = 183.4$ ms and $E[T_P] = 793.6$ ms, respectively.

This kind of study illustrates how important system behavior characteristics are in EHSNs, which are not obvious at all in their real-life working conditions. In particular, this example shows the large impact the residual time has in the residence time of the highest priority task (T_S). Indeed, when a high-priority task arrives to the system, the microcontroller can be fully occupied by low-priority tasks, and the new high-priority task has to wait until the microcontroller becomes free, which can cause serious problems of accuracy in the damage estimation done be Shimmer if there is no energy harvested

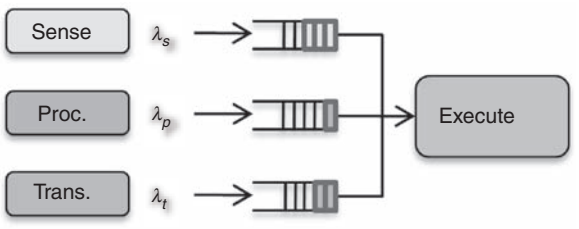

Figure 6. *Residence time example.*

Table 1. Priority task execution example data.

	λ (req/s)	$E[\tau]$ (ms)	P (mW)
Sense	0.1	1.1	1027
Process	0.7	510	680
Transmit	0.1	47.1	165

to execute the new task after the low-priority ones have finished. Thus, reducing the time that low-priority tasks occupy the microcontroller, as HOLLOWS targets, subsequently reduces the residence time of high-priority tasks.

IMPACT OF THE ENERGY HARVESTING UNIT

Once we have all the needed information to compute the total estimated energy needed to execute a new task T_k that enters the system, $E[E_k]$ of Equation (1), the needed energy to execute all the tasks of higher or equal priority waiting in the system can be calculated as:

$$E\left[J_k^W\right] = \sum_{i=1}^{k} E[J_i]E\left[N_{qi}\right] = \sum_{i=1}^{k} E[J_i]\lambda_i E[W_i], \qquad (11)$$

where $E[J_i]$ is the expected energy consumption of a task of type i and $E[N_{qi}]$ is the number of tasks of type i already waiting to be executed in the queue. Then, the number of tasks waiting in the system can be calculated in real time, or estimated a priori by applying Little's formula (Leon-Garcia, 2008), $E[N_{qi}] = \lambda_i E[W_i]$.

The number of higher priority tasks that arrive during the waiting time $E[J_k^A]$) can be computed as:

$$E\left[J_k^A\right] = \sum_{i=1}^{k-1} E[J_i]E[M_i] = \sum_{i=1}^{k-1} E[J_i]\lambda_i E[W_k], \qquad (12)$$

where $E[M_i]$ is the expected arrival of tasks of higher priority, that is $E[M_i] = \lambda_i E[W_k]$.

Finally, the residual energy needed to finish the task found in service can be calculated as:

$$E\left[R^J\right] = 1/2 \sum_{i=1}^{n} \lambda_i E\left[J_i^2\right]. \qquad (13)$$

At this point, we are able to compute $E[E_k]$ only knowing the expected queue waiting time, the arrival rate of the different types of tasks and the expected energy consumption of T_1 to T_n.

Returning to the previous example of a Shimmer setup, described in Table 1, we can now calculate the expected amount of necessary energy to finish executing a new task. Using Equations (11), (12) and the waiting time computed for this example, that is, $E[W_k] = (182.3, 283.6, 444.3)$ ms, we obtain $E[J_k^W] = (20.6, 68.9 \times 10^3, 69.2 \times 10^3)$ μJ and $E[J_k^A] = (0, 32, 107.9 \times 10^3)$ μJ. Then, using the modeling of the execution time as an exponential random variable (see section 'Delay in Priority Queue Systems'), we obtain $E[R^J] = 123.8$ mJ. Finally, the total energy needed will be $E[E_k] = (125, 539.5, 308.7)$ mJ.

At this point, we are able to compute how the energy harvesting unit of the EHSN (e.g., a solar panel in of Shimmer) affects the execution waiting time of the tasks in the system. The waiting time for a task T_k, entering the system at time t, should not be affected by the harvesting unit if the energy needed to execute T_k ($E[E_k]$, see Equation (1)), is less than the energy already present in the system when T_k arrives (E_t), plus the energy the system is able to store during the waiting time:

$$E[E_k] \leq E_t + E[T_k](\eta E_{\text{enter}} - E_{\text{leak}}), \qquad (14)$$

where η is the storage efficiency of the energy harvesting unit, E_{enter} is the expected energy entering the system in the near future and E_{leak} is the energy leakage of the node. If this inequality is true for all T_k entering the system, the mean waiting time is only dependent on the arrival rate and mean execution time.

However, there is not always enough energy to execute a new task that arrives into the system. Thus, the waiting time of an incoming task will be affected by the harvesting system, as the task will remain in the system until enough energy is harvested from the solar panel. Hence, this waiting time can be computed using Equation (14), as follows:

$$E[T_k] = \frac{E[E_k] - E_t}{\eta E_{\text{enter}} - E_{\text{leak}}}. \qquad (15)$$

Finally, according to our working setup of Shimmer, defined in the previous section, we need to execute a *Transmit* task a total energy ($E[E_T]$) of 308.7 mJ. Hence, supposing that there is at a certain moment in time only $E_t = 100$ mJ (i.e., less than 3% of the energy in the Shimmer storage unit (Musiani et al., 2007)), and assuming that the store efficient is $\eta = 0.7$ and ρ_{leak} is null, the expected waiting time will fully depend on the harvesting unit. Hence, if the power entering the system in the near future is $\rho_{SP} = 230$ mW (i.e., a typical energy income at noon for a cloudy day), then the expected residence time will be $E[T_T] = 1296.3$ ms, which is 2.6× larger than the residence time computed in the case that enough energy exists in the system (see section 'Delay in Priority Queue Systems'). Hence, it is a must to have efficient algorithms to suitably estimate the forthcoming energy harvesting income, so that the queuing model can accordingly schedule the tasks to be executed in EHSNs.

As a consequence, in the next section we introduce a new energy harvesting prediction algorithm targeted for EHSN architectures, which can estimate the short- and long-term energy availability in the nodes and, as a result, the HOLLOWS scheduler can perform a accurate estimation of the worst waiting time of queued tasks of different priorities to be executed in the future.

Weather-conditioned Moving Average Energy Prediction

The WCMA energy harvesting prediction algorithm estimates the energy entering the system based on sampling the power delivered by the solar panel. WCMA has its foundations on the EWMA (Cox, 1961). EWMA characterizes the seasonal changes by adapting both the change in the hour of sunrise and sunset as well as the difference in solar power income between seasons. However, in contrast with EWMA, the new WCMA algorithm takes also into account sudden weather changes with minimal overhead and prediction error.

In EWMA, the day is divided in slots and a vector of estimated values for each slot i is stored, that is, $X(i)$. This equation is used to update the slots, as follows:

$$X(i) = \alpha \cdot X(i-1) + (1-\alpha) \cdot x(i), \tag{16}$$

where $x(i)$ denotes the value of real power observed at the end of the slot i and α is a weighting factor.

Figure 7 shows the actual power input from the Shimmer solar panel and the predicted value in five consecutive days, with a mix of sunny and cloudy conditions, for an optimized value of $\alpha = 0.5$ (Kansal et al., 2007). In this case, when the sunny and cloudy days alternate, the EWMA produces a significant error in its prediction, due to the high impact of the solar conditions of previous day in the predicted value. To avoid this effect, our new prediction algorithm takes into account not only the solar conditions at a specific time of the day, but also the weather conditions in the current day. This is especially important in frequently changing weather conditions, for example, as it has been observed in typical solar traces (Raghunathan et al., 2005), the energy harvested during cloudy days can be less than half of the value gathered during sunny days.

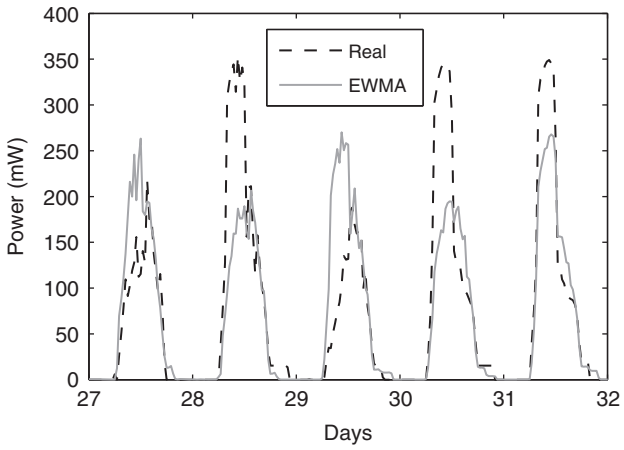

Figure 7. *EWMA power prediction algorithm.*

Our WCMA algorithm uses a P matrix of size $D \times N$ that stores N sampled input power values per day from the solar panel for each D past days. Hence, $P(d, n)$ is the stored power in the matrix that corresponds with the input measured power within the d-th day, at the beginning of the n-th time slot. Then, the predicted constant input power for the n-th time slot of the current day $P_{\text{Pred}}(n)$ is related to the previous power samples in the same day and the mean value of the past samples (at the same hour of the day):

$$P_{\text{Pred}}(n) = \alpha \cdot P(d,n) + GAP_K \cdot (1-\alpha) \cdot M_D(n+1), \tag{17}$$

where α is a weighting factor similar to the EWMA algorithm, and $M_D(n+1)$ is the mean of D past days at the $n+1$ sample of the day:

$$M_D(n) = \frac{\sum_{i=1}^{D} P(i,n)}{D}. \tag{18}$$

The main innovation in our algorithm is the inclusion of the GAP_K factor. This factor measures the solar conditions within the current day with respect to the previous days. To compute the GAP_K factor, we first define a vector $\mathbf{V} = [v_1, v_2, \ldots, v_K]$ with K elements. $\mathbf{V}$ contains the quotient of the past K samples and the average solar power available during the previous D days for those samples. Therefore, a value greater than one means that today's values are larger than the mean, which consequently represents a sunny day and, similarly, values smaller than one represent cloudy days:

$$v_k = \frac{P(d,n-K+k)}{M_D(n-K+k)}. \tag{19}$$

Then, in order to give more importance to the closest values on time, we weight these values with the distance to the actual point in time using vector $\mathbf{P} = [p_1, p_2, \ldots, p_K]$, as follows:

$$p_k = \frac{k}{K}. \tag{20}$$

Finally, the weighting factor, GAP_K, is computed:

$$GAP_K = \frac{\mathbf{V} \cdot \mathbf{P}^T}{\sum \mathbf{P}}. \tag{21}$$

Using the real-life working conditions and characteristics of the solar panel included in the Shimmer node (Musiani et al., 2007), Table 2 shows an example of how WCMA computes the GAP_K factor to compute the

input power prediction for the n-th time slot of the current day $P_{\text{Pred}}(n)$ with $D=4$, $K=3$. The M_D vector contains the mean value of the previous 4 days, V has the quotient of the elements in row d divided by the M_D vector (element by element), and P is the weighting factor for V. Finally, the GAP_k value is computed as follows:

$$GAP_k = \frac{(1.12,0.83,0.75)\cdot(0.33,0.67,1.00)^T}{\sum(0.33,0.67,1)}. \quad (22)$$

And the predicted value with $\alpha=0.7$ is:

$$P_{\text{Pred}}(d,n) = 0.7\cdot 230 + 0.84(1-0.7)\cdot 313 = 206. \quad (23)$$

WCMA'S PARAMETERS OPTIMIZATION

To optimize all the needed parameters in WCMA, that is, the size of the E matrix ($D\times N$), the α factor and number of past samples to weight (K), we must define the error function to evaluate relevant constraints. To optimize these values we have recorded the available power from the solar panel (PowerFilm, 2009) of the Shimmer node every minute during 45 consecutive days. As we would like to predict the Sun evolution, the night values are discarded in the computation of the error. Thus, we consider as night values all the samples with less than 10% of the maximum figure. Then, the error function for a record of N points is given at the percentage:

$$\text{Err} = \frac{1}{N}\sum_{i=1}^{N}\text{abs}\left(1-\frac{E_{\text{Real}}}{E_{\text{Pred}}}\right), \quad (24)$$

where E_{Real} denotes the real energy value during the time slot, obtained by integration of the recorded power values, and E_{Pred} is therefore the estimated value.

To optimize the WCMA predictor performance, we do not only focus on minimizing the error, but we also explore possible trade-offs between accuracy and duty cycles. Indeed, when more samples are collected per day, the following energy harvesting estimate is more precise, but more frequent sensor node wake-ups are needed, which can lead to a negative impact on the overall energy consumption. On the other hand, a very low sampling rate may not give the sufficient tuning data to our WCMA predictor. Thus, it will not be able to estimate the energy harvesting rate, which would make the sensor node calibration difficult. In this regard, based on our performed experiments on Shimmer and other solar-panel-based real sensor nodes (Musiani et al., 2007), a sample period of 30 min (i.e., 48 samples/day) gives a reasonably accurate prediction with a low duty cycle and a small memory footprint. In this regard, Figure 8 shows the estimated error of the prediction as a function of the weighting factor and the number of days (D), for a fixed and small number of past values ($K=6$) and samples per day ($N=48$), which implies a very limited power consumption to tune our WCMA predictor for the HOLLOWS scheduler. Selecting a weighting factor (α) of 0.7 gives a minimal error, independently of the number of past days stored in the matrix. Hence, we have used this value in our optimization process of the WCMA predictor, running as part of our HOLLOWS task scheduler for EHSNs.

Then, we have tuned the rest of the parameters in the WCMA predictor according to the experimental data for our Shimmer case study (other EHSN platforms would follow a similar optimization process, which only requires few minutes of simulations). Figure 9 shows the prediction error versus D and K, with fixed α and N. Our experiments indicate that, if the number of past samples K is above 5, then the error quickly increases because it takes into account too many samples of the weather pattern of each day. Since the number of past days does not influence the error as much as the number of past samples K for a particular

Table 2. Solar power prediction example.

	Solar panel power evolution in mW			
	$n-2$	$n-1$	n	$n+1$
$d=4$	277	272	221	263
$d=3$	350	353	347	347
$d=2$	345	346	349	353
$d=1$	249	255	314	289
$d=0$	342	256	230	
Mean	305	306	307	313
V	1.12	0.84	0.75	
P	0.33	0.67	1.00	

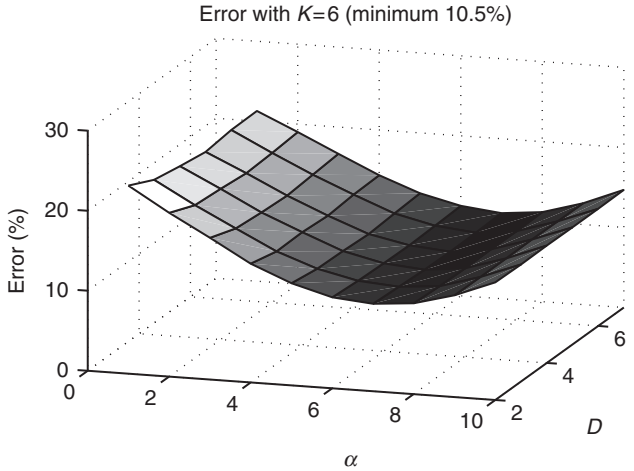

Figure 8. *Estimated error for N=48 and K=6.*

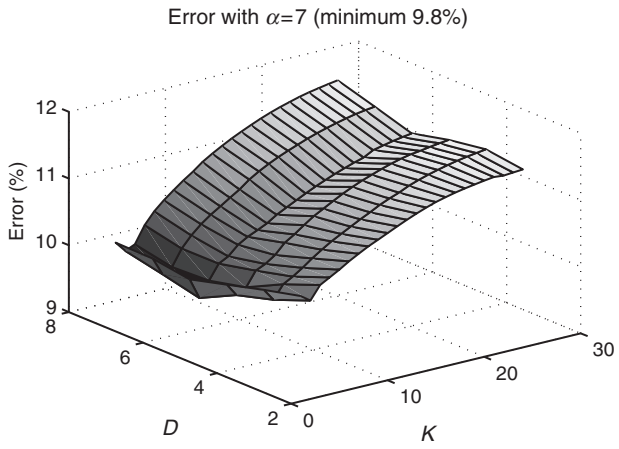

Figure 9. *Estimated error for N=48 and α=7.*

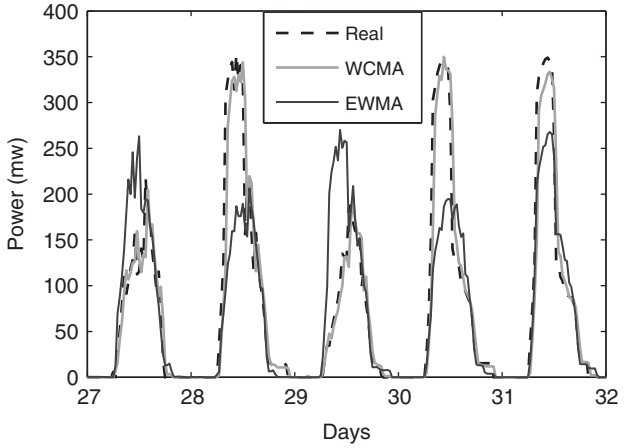

Figure 10. *Energy harvesting prediction accuracy of WCMA vs EWMA.*

day, then we can use fewer days for the estimate, which lowers the computational cost of WCMA without a significant accuracy loss. As a result of our analysis, the WCMA predictor used as part of the HOLLOWS task scheduler has the following parameters to minimize its prediction error: $D=4$ days, $N=48$ samples/day, $K=3$ past samples, and $\alpha=0.7$.

ENERGY HARVESTING PREDICTION COMPARISON OF WCMA VERSUS EWMA

Once we have tuned both the WCMA and the EWMA energy harvesting predictors, we have compared the energy prediction accuracy of both of them in various real-life working conditions of the Shimmer node. Figure 10 shows five consecutive days of different weather conditions and the predicted values using both algorithms. The 27th and 29th days correspond to cloudy conditions and the rest of the days are sunny. Since EWMA only uses values from previous days at the exact same time period, if the weather conditions change from one day to another, this method has a large error in prediction (i.e., close to 50%, as shown in the 30th day). On the other hand, WCMA produces a much better prediction because it uses the values from the same hour over a number of previous days and the past values from the same day, which help to calibrate the estimation against the actual weather conditions. Overall, EWMA gives an average error of 28.6% compared to 9.8% obtained by our new algorithm WCMA (i.e., 3× less average prediction error for our algorithm), over the complete set of 45 days of the collected solar panel data in the Shimmer node.

EXPERIMENTAL SETUPS OF EHSNs AND RESULTS

We have assessed the quality of our power-aware task scheduler, HOLLOWS, in different working scenarios of the Shimmer EHSN case study (described on the section 'Architectures of Energy Harvesting Sensor Nodes'). To this end, we have created a Matlab model and a complete EHSN multi-queuing simulator for HOLLOWS running in the Shimmer node. Furthermore, this validation and simulation environment can be applied to other EHSN architectures by just modifying the HOLLOWS input specifications of the final working conditions of the target node/s (see section 'Architectures of Energy Harvesting Sensor Nodes').

In particular, the scenario that will be used to illustrate the benefits of the HOLLOWS scheduler is the following:

- The mean execution time and energy consumption for the *Actuate/Sample (A)*, *Process (P)* and *Transmit (T)* tasks are fixed, as indicated in Table 1, based on those of Shimmer (Musiani et al., 2007), and identifying *Actuation* & *Acquisition* with a single task *A*.
- Three tasks priorities are defined in the system, namely, the highest priority task is actuate (*A*), the medium priority task is process (*P*) and the low priority one is transmit (*T*).
- The optimization strategy performs the structural health monitoring process trying to achieve the highest possible accuracy, according to the available energy and the predicted energy harvesting income.

Regarding the level of damage estimation accuracy in structural health monitoring, as we have already mentioned in the section 'Architectures of Energy Harvesting Sensor Nodes', to accurately locate and identify the severity of any structural damage, multiple sensing, filtering, and identification algorithms must be executed in the Shimmer node using different signals to excite the PZTs. Each of the different types of

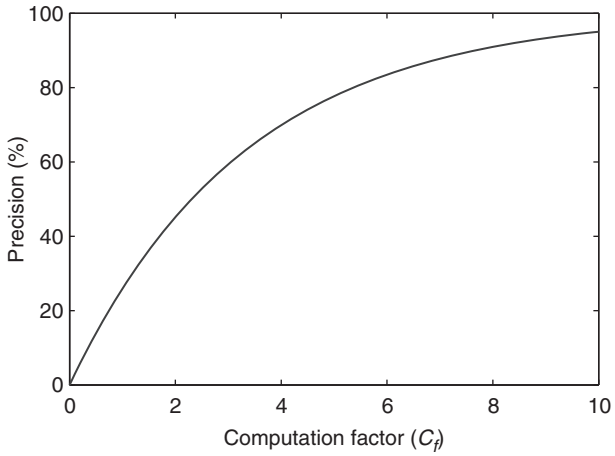

Figure 11. *Relative accuracy for Actuate/Sample and Processing tasks relative to Computation Factor in structural health monitoring.*

statistical modeling algorithms (i.e., group classification, regression analysis and outlier detection) have their advantages and disadvantages, but to maximize the accuracy of the results, in all the cases it is needed to increase the microcontroller processing time to filter, cleansing, and perform feature evaluation. This increase in processing time results in an increase of accuracy of the damage detection (as long as the microcontroller processing workload does not go beyond 100% utilization) (Farrar and Worden, 2007; Bradely Steck and Rosing, 2009). Then, regarding data acquisition, the *Actuate*/*Sample* process for Shimmer consists of generating and sensing vibrations along the structure, and due to the nature of the process, it can be affected by different factors (e.g., wind gusts, vehicles crossing a bridge, elevators in a building, etc.). Therefore, to increase the accuracy of the damage detection process, multiple processing (P) tasks can be executed in the same path to reduce these interference factors and increase the signal-to-noise ratio.

A single P task can be identified with the microcontroller processing time needed to compute a fast fourier transform (FFT) of the complete data recorded during a single A task. In fact, Musiani et al. (2007) identify a complete microcontroller processing analysis of one path with seven P tasks, each of them represents the energy needed to execute an FFT. An extra P task has been included in our tasks execution setup for structural health monitoring to average the sensed data. Finally, the *Send* task can be identified with the energy consumption of sending the filtered values obtained from sensing one path, which is 10× less than the raw sensed data. Therefore, in order to analyze the impact of multiple tasks executions of each type in the accuracy results, we have defined a general computation factor (C_f). This factor relates the accuracy of the results with the energy used in the A and P tasks (Figure 11), and it is an abstraction that associates a certain level of accuracy with a computation effort. Thus, we have identified C_f as the number of A and P tasks executed for each path. In general, the more energy and CPU time a node uses for structural damage detection, the higher the achieved precision is, but up to a maximum level associated to each applied detection technique. Based on typical experimental setups of the Shimmer node, we have chosen an inverse exponential function to model the scenario in which an initial small increase in the energy used for the damage detection, causes a significant increase in precision. For example, shifting form $C_f = 1$ to $C_f = 2$, the precision figure doubles (from 10% to 20%). However, beyond a certain saturation point, increasing C_f has almost no impact in terms of accuracy increase, for example from $C_f = 8$ to $C_f = 10$ the damage precision increase is almost negligible (less than 5%), but causes a large increase on spent time and energy during the process (25% more for both).

As mentioned before, structural health monitoring can be separated into two basic categories: periodic lifetime monitoring and rapid event assessment. Therefore, the periodic lifetime monitoring that identifies accumulative damage over a long period of time will be used in the following section to validate HOLLOWS simulator for the Shimmer node. Then, the rapid event assessment context, which is needed to obtain data from a structure immediately following a significant event (earthquakes, fires, accidents, etc.) will be used as a case study to explore the capabilities of HOLLOWS to achieve different punctual trade-offs between energy, time and accuracy in real-life Shimmer working conditions.

Assessment of the HOLLOWS Task Scheduler for Periodic Lifetime Monitoring

To validate the HOLLOWS scheduler, a periodic lifetime scenario of structural health monitoring has been studied using the HOLLOWS multi-queue simulator and a Matlab analytical model with the equations of the section 'Task Scheduling for Energy Harvesting Sensor Nodes', both using the Shimmer node setup described in the previous section.

HOLLOWS OPERATION WITHOUT ENERGY CONSTRAINTS

In our first set of experiments, we have compared (Figure 12) the residence time $E[T_k]$ values obtained by the HOLLOWS simulator and with the analytical values derived from the analysis presented in the section using an infinite energy scenario. The system has been configured to operate at 90% of structural damage estimation accuracy ($C_f = 8$). Since we do not set any energy constraints, we can explore the Shimmer node operation capabilities. In this context, the node is able to perform a complete scan of the surface every 50 min, using three

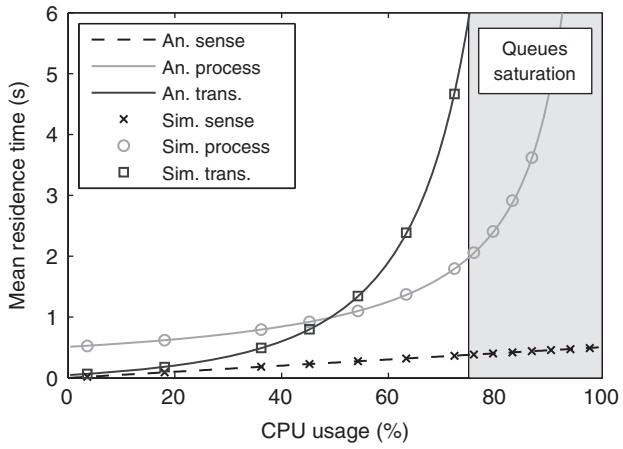

Figure 12. *Mean residence time comparison for three prioritized tasks between analytical model and HOLLOWS simulation in Shimmer.*

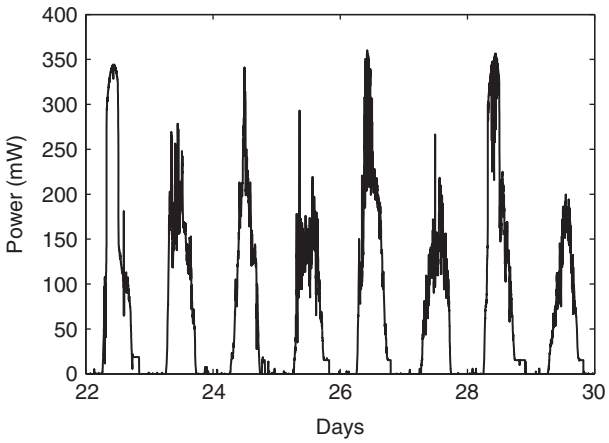

Figure 13. *Solar panel power evolution sample of sunny and cloudy mixed days.*

different feature extraction algorithms per path, one of each kind of the three statistical modeling types (i.e., group classification, regression analysis and outlier detection).

In addition, as depicted in Figure 12, the region above 75% of DSP microcontroller utilization is labeled as a *Queues Saturation* zone. A queues saturation scenario occurs when the residence time of low priority tasks increases exponentially with respect to the microcontroller utilization due to the queue waiting time, that is, a *Sense* task has to wait 1.55 h in its queue to be executed on a 99.5% microcontroller utilization scenario. Also, high-priority tasks have a large increase of residence time due to the residual service time, namely, from 20 to 500 ms (Equation (7)), because most of the time the microcontroller is busy and HOLLOWS uses a non-preemptive scheduling in the Shimmer node. Thus, the current task being executed cannot be removed from the microcontroller until its execution has finished.

The difference between the analytical and the simulation framework of the HOLLOWS scheduler, with respect to the residence time using the Shimmer setup, is less than 1% in mean value for the three tasks in the whole sweep of microcontroller utilization. Thus, these experimental results prove the correct analytical foundations of using multi-queuing theory for the design of our HOLLOWS task scheduler targetting EHSNs.

HOLLOWS SCHEDULING UNDER ENERGY HARVESTING INCOME CONSTRAINTS

To introduce the energy constraints of an energy harvesting device into the Shimmer setup where HOLLOWS is executed, we have employed a record of the energy delivered by its solar panel during 45 consecutive days. In general, the Shimmer solar panel is 150 cm^2 and can deliver up to 360 mW (PowerFilm, 2009). Then, Figure 13 shows a sample of this solar

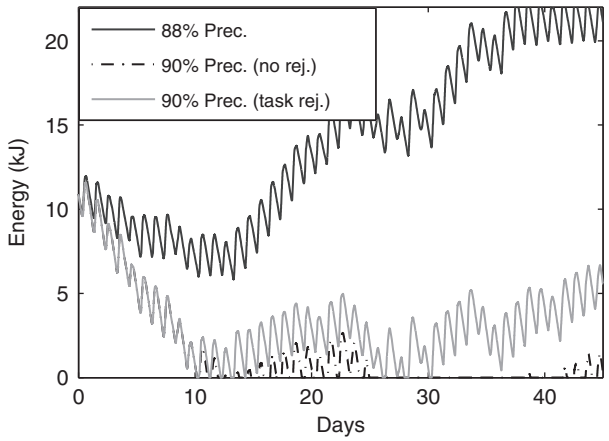

Figure 14. *System energy evolution with self-adaptation of damage precision quality disabled.*

panel data record with a mix of sunny and cloudy days, which has a mean value of 76 mW.

In order to guarantee long-time sustainable operation, Kansal et al. (2004) propose a simple model for EHSN that ensures energy neutral operation (ENO). ENO is a state where the node does not consume more energy than what is able to harvest, and consequently it is able to operate forever. Using this model in Shimmer with the considered solar panel record, the storage unit of the Shimmer node has to be at least 22 kJ, with an initial amount of 11 kJ, and the mean energy consumption being equal or less than 53 mW[2].

In this regard, Figure 14 shows the system energy evolution in a situation in which a complete structure scan has to be done every 10 h using three different feature extraction algorithms per path. In the first case, the computation factor (C_f) requested to the HOLLOWS

[2]All these values assume an energy storage efficiency of 70%, average figure in today's storage units of EHSNs, and leakage figures can be neglected.

Table 3. HOLLOWS tasks execution results in the periodic lifetime scenario.

	Analytical			Simulation			Error	Rej.
Accuracy	*A*	*P*	*T*	*A*	*P*	*T*	$E[T_k]$	tasks
88% ($C_f = 7$) Energy 90%	38 ms	549 ms	89 ms	37 ms	548 ms	90 ms	0.9%	0%
90% ($C_f = 8$) Energy 105%	43 ms	556 ms	97 ms	404 s 41 ms	273 min 613 ms	51 h 97 ms	Exp. 4.5%	0% 2%

scheduler has been fixed to 7, corresponding to a required accuracy level of 88%. In this situation, the mean energy usage is less than 100% of the harvested energy, thus, the battery level reaches the maximum capacity (22 kJ), which implies that the system wastes energy but this situation does not affect the tasks execution delay. In the second case, C_f has been set to 8, that is, a 90% required accuracy level is requested to our HOLLOWS scheduler. However, in this situation the energy usage is about 105% of the harvested energy. Thus, if HOLLOWS cannot modify dynamically the requested accuracy to adapt the task execution order to the incoming energy harvesting level, the system will run out of energy and the level of accuracy cannot be guaranteed. In fact, this is what would occur with state-of-the-art EHSNs fixed-priority schedulers (Lin et al., 2005; Magno et al., 2008), which would make the mean residence time of tasks in the EHSNs increase exponentially. However, HOLLOWS has been designed specifically to dynamically adapt the order of the execution of the tasks to avoid this situation. This adaptation is based on its multiple-priority queues and precise WCMA prediction model for energy harvesting income. Thus, within this working setup of Shimmer, as shown in the last case of Figure 14, HOLLOWS reduces slightly at run-time the actual returned damage estimation precision of the node by rejecting a very reduced amount of certain incoming tasks (i.e., rejecting less than 2% of tasks for each type), which reduces less than 4% the overall damage accuracy estimate but, at the same time, HOLLOWS is able to reach the desired ENO state for the Shimmer node. Hence, the node is able to operate continuously during the whole period of 45 days only with a small reduction with respect to the originally desired accuracy (which cannot be reached in any case with the actual energy harvesting income level in the 45 days).

This adaptation of damage accuracy level is performed by HOLLOWS as introduced in the section 'Delay in Energy Harvesting Systems', namely, before adding a new task to the queue, HOLLOWS compares the actual energy with the expected energy needed to execute the task in the multi-priority queuing model, without harvesting delay, and rejects adding a new task if there is not enough energy after considering also the estimation of the WCMA prediction model for the energy harvesting income. Moreover, if the number of rejected tasks reaches a certain configurable tolerance value, HOLLOWS can dynamically reconfigure the computation factor (C_f) to meet the energy constraints. Thus, it can achieve the maximum possible accuracy according to the predicted energy harvesting income. Indeed, Table 3 shows that HOLLOWS incurs very low residence time estimation mean errors, which proves the accuracy of the overall approach to guarantee ENO operation in real-life working setups of EHSNs.

In addition, by exploiting the adaptation capabilities of HOLLOWS we can guarantee a minimum quality level in the results provided each day, based on the minimum energy income during a cloudy day. Furthermore, HOLLOWS will dynamically adapt this minimum requested level each day using the WCMA energy estimator to increase the precision for sunny conditions. To this end, HOLLOWS starts operating with the assumption that all the days are cloudy and, in case of a better situation, the excess of energy income from the energy harvesting devices is used to dynamically increase the quality of the results. In this context, Figure 15 shows the precision per day of the structural health monitoring process using the application of the periodic checks of the WCMA predictor included in HOLLOWS. As this figure illustrates, a minimum precision of 70% is guaranteed by HOLLOWS for the defined Shimmer setup, which corresponds to days 27 and 29, and up to 93% will be dynamically obtained during very sunny conditions, which occur, for instance, in days 28, 37, or 41.

Assessment of the HOLLOWS Task Scheduler for Rapid Event Assessment

Rapid event assessment operation in structural health monitoring is needed to obtain data from a structure immediately after a potentially catastrophic event (e.g., earthquake, fire, accident on the monitored structured, etc.). Since this is a very special situation, high precision damage estimation is required in a certain instant of time, rather than continuous (and regular) monitoring, as in the case of periodic lifetime structural health monitoring. For instance, in the rapid event assessment mode, instead of ensuring ENO, a Shimmer node may be requested to use all its available energy and predicted energy harvesting income to perform a complete

structural test with the highest possible accuracy. Nonetheless, the number of tasks that can be executed during a certain interval must respect the constraint of not going beyond 100% utilization of the microcontroller workload capabilities, as indicated in the following equation:

$$U_{CPU} = \sum_k \mu_k = \sum_k \lambda_k E[\tau_k] \leq 1. \quad (25)$$

Furthermore, also tasks delays have to be considered, which can become a serious issue. Indeed, Figure 12 shows how the residence time increases exponentially if the microprocessor utilization is close to 100%, that is, more than 1 h of residence time is estimated for the *Sense* task. Evidently, these large residence time values are not acceptable in this rapid event assessment scenario and our proposed HOLLOWS task scheduler can be used to exploit dynamically the existing tradeoffs between tasks delay, energy consumption, and accuracy to avoid this situation.

As a result, in this set of experiments we focus on optimizing the energy parameter in the Shimmer node using HOLLOWS. In this regard, HOLLOWS relies once again in the novel WCMA energy harvesting estimator algorithm, which is able to accurately predict the time window in which the system will harvest the biggest amount of energy as well as estimating its total amount. Thus, using this value and the available energy in the storage unit, HOLLOWS will schedule the execution of the queued tasks that maximize the accuracy of the structural damage estimation, estimating at the same time which will be the waiting time for the chosen tasks to be executed. To this end, HOLLOWS can be configured for a certain worst response time in case of an emergency situation by defining a maximum task waiting time. Then, HOLLOWS will accordingly decide the set of queued tasks that can be executed at the precise moment of each emergency. Finally, HOLLOWS can also be used to guarantee a minimum energy level that needs to be saved in order to be used only during emergency situations, such that we can ensure a minimum quality of the structural check within the emergency context.

To illustrate these principles, we consider the case of an emergency event (e.g., an earthquake). Since Shimmer has a total of 16 PZTs (i.e., a total of 240 different paths can be sensed), if we define a total time window of 30 min to send all the possible results in an event assessment, HOLLOWS takes multiple decisions to optimize the quality of these results. Figure 16 shows the energy and microcontroller utilization usage versus the estimated damage detection precision performed by HOLLOWS for three different cases, namely, a one-, two- or three-feature extraction algorithm used for each path. As this figure depicts, HOLLOWS will not increase the accuracy beyond a certain level (no matter if a higher quality has been requested to the Shimmer node), if the microcontroller utilization is already 100%, so that a guaranteed execution delay for all the tasks can be estimated. Moreover, HOLLOWS will warn the user if the scheduled tasks tries to operate the Shimmer beyond 75% microcontroller utilization, since it will detect the saturation level of the queues using its internal estimated waiting time, based on the multiple priorities queues model (see section 'Task Scheduling for Energy Harvesting Sensor Nodes' for more details) and report the high residence time of the tasks.

In addition, Figure 16 shows the three curves, using the 1–3 feature extraction procedures, which illustrate how HOLLOWS will choose at run-time between the execution of the three different algorithms per path, according to the requested damage detection precision and the expected energy harvesting income predicted by its internal WCMA predictor. In the first case, HOLLOWS is able to provide a good compromise

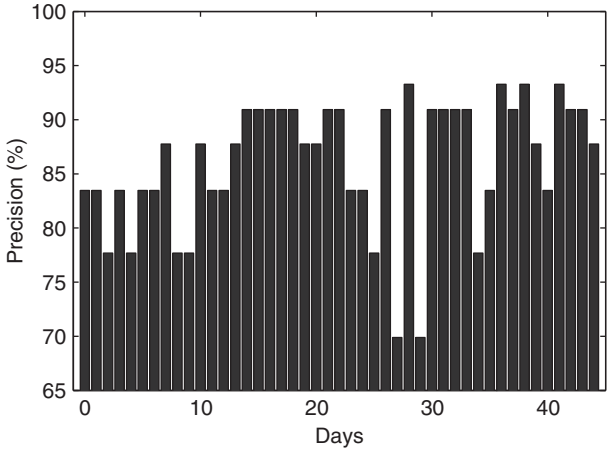

Figure 15. *Periodic lifetime structural health monitoring precision over days.*

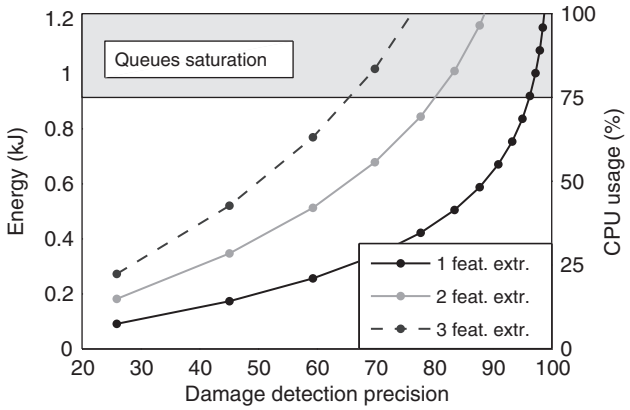

Figure 16. *Energy and CPU usage vs damage detection precision.*

between energy, time, and accuracy, by using a C_f of 4, which implies an accuracy of 70% with a total energy consumption of 1 kJ approximately. As a result, the system works on the border of a queue saturation mode, and the residence time for the *Transmit* task is 14 s, which can be admissible in this 30-min emergency scenario. In the second case, if the WCMA energy harvesting predictor included in HOLLOWS estimates an energy income in the 30-min period that (combined with the currently available energy stored in Shimmer) allows to consume up to than 1.2 kJ of energy, then HOLLOWS would schedule the execution of two-feature extraction algorithms, so that the Shimmer node can achieve a higher precision (83% or $C_f = 6$). As in the previous case, the system would work on queue saturation mode, and very close to the 100% microcontroller workload limit, but still with a residence time for the *Transmit* task of 13 s, which is still acceptable in a 30-min worst response time scenario. In the third case, in case the highest possible accuracy is requested and no constraint in the task waiting time is set, HOLLOWS would execute one feature extraction per path, which will need only 0.9 kJ to execute the algorithm with an accuracy of 96% ($C_f = 11$), which would be again placing the working point of Shimmer in low queue saturation mode. However, in this situation (since the residence time was not defined to any real constraint but a very high value), the *Transmit* task would have a waiting time of 92 ms. Furthermore, HOLLOWS would not choose any higher accuracy point because very small gains in accuracy would be achieved, while the task waiting time would increase exponentially.

Finally, in the last set of experiments we have assessed the importance of having an accurate energy harvesting estimator in EHSNs (such as our proposed WCMA predictor of section 'Weather-conditioned Moving Average Energy Prediction') for the design of our power-aware tasks scheduler, that is, HOLLOWS. In this regard, Table 4 shows the estimated energy using either EWMA or WCMA in HOLLOWS versus the real amount of available energy to harvest during a time window of 30 min at noon for four consecutive mixed days. As this table shows, the accurate energy harvesting prediction algorithm we have included in HOLLOWS is crucial for EHSN systems, in order to optimally exploit the energy income in real working conditions of multi-priority task sensor nodes. In fact, as Table 4 depicts, WCMA has a maximum energy prediction error of merely 10%, while EWMA can have errors of up to 90%. Hence, the large EWMA energy estimation error can seriously damage the obtained accuracy level of the structural health monitoring process due to incorrect energy management decisions applied in Shimmer. In particular, if we consider days 7 and 9 in Figure 15, both WCMA and EWMA achieve similar (excellent) prediction results, because the energy income estimation and the achievable maximum accuracy level are matched. However, in day 8, on the one hand, the WCMA algorithm included in HOLLOWS correctly estimates the energy income and the right set of queued multi-priority tasks to be executed is selected. Thus, Shimmer can execute one feature extraction algorithm with a 25% of accuracy or two with 45%, as shown in Figure 16, just using the energy income from the solar panel. On the contrary, EWMA overestimates the energy income in day 8 by almost 0.26 kJ, which leads the HOLLOWS scheduler to incorrectly increase the target accuracy value of the executed feature extraction algorithms. Thus, the Shimmer node would run out of energy in the middle of the process and no guarantee of the accuracy of the predicted damage can be given. Similarly, during day 10, HOLLOWS correctly estimates the maximum accuracy achievable using our included WCMA algorithm (i.e., one feature extraction process with 45% accuracy level, two with 60% or three with 88%), while EWMA underestimates the energy income and drastically reduces the accuracy to only one feature extraction with 25% accuracy or two damage analyses with 45%. Thus, these results indicate the benefit of the proposed combination in our HOLLOWS task scheduler of the novel WCMA prediction algorithm and the prioritized task multi-queuing system in EHSN platforms, in order to achieve an excellent overall energy harvesting management (i.e., more accurate results of the monitored environment for the same energy harvesting income pattern).

Table 4. Solar energy prediction results.

Day	E_{real}(J)	Algorithm	E(J)	Err%
7	571.72	WCMA	550.44	3.72
		EWMA	535.50	6.34
8	284.63	WCMA	255.60	10.20
		EWMA	543.60	−90.99
9	400.61	WCMA	360.00	10.14
		EWMA	423.00	−5.59
10	609.50	WCMA	597.60	1.95
		EWMA	406.80	33.26

CONCLUSIONS

EHSN are a promising technology for autonomous monitoring of environmental events. However, their very low-energy consumption requirements pose great challenges to design EHSNs that must concurrently run multiple types of tasks (signal processing, communication, actuation, etc.), as no accurate models exists to adjust their workloads to variable weather conditions and dynamic set of tasks (with different priorities). In this article we have proposed HOLLOWS, a new power-aware task scheduler for energy harvesting-based sensor nodes. It includes an accurate and fast analytical

solution based on prioritized queues to predict at run-time the mean waiting and residence time for different multi-level priority tasks in EHSNs. Our proposed task scheduler only requires as input the inter-arrival time and expected execution time for each task, the available energy in the node and an estimation of the expected energy income in the future. In this regard, HOLLOWS incorporates a new solar prediction algorithm, WCMA, specifically targeting solar-panel-based EHSN. Our experimental results in real-life working setups of the Shimmer node, a last-generation structural health monitoring EHSN, have shown that HOLLOWS is able to maximize the use of the dynamically varying solar panel power income with an estimation error of tasks waiting time of 1%. As a consequence, the accuracy of the energy harvesting prediction achieved by HOLLOWS is up to 3× better than state-of-the-art energy prediction algorithms for EHSNs, such as, EWMA.

In addition, our proposed power-aware scheduler is able to operate in a scenario in which the energy demanded by the initially requested tasks by the user monitoring the system is higher than the energy entering the system. In this case, HOLLOWS informs the user and rejects only the minimum number of tasks to ensure the expected residence time and to guarantee a minimum level of accuracy for the available and expected harvested energy. Finally, HOLLOWS can also be used to schedule tasks using dynamic thresholds for the target residence time and energy constraints, enabling run-time adaptation of the Shimmer node case study (in the order of seconds) for unexpected emergency conditions (e.g., high-precision damage assessment in case of earthquakes, flooding, etc.)

ACKNOWLEDGMENTS

This research has been partly supported by the Swiss NSF, Grant number 20021-127282. This work is also partially supported by the Spanish Government Research Grants TIN2008-00508 and CSD00C-07-20811.

REFERENCES

Benini, L., Bogliolo, A. and De Micheli, G. 2000. "A Survey of Design Techniques for Systemlevel Dynamic Power Management," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 8:299–316.

Bradely Steck, J. and Rosing, T. 2009. "Adapting Performance in Energy Harvesting Wireless Sensor Networks for Structural Health Monitoring Applications," In: *Proceedings of International Workshop on Structural Health Monitoring (IWSHM 2009)*, Stanford, California, USA.

Cox, D.R. 1961. "Prediction by Exponentially Weighted Moving Averages and Related Methods," *Journal of the Royal Statistical Society. Series B (Methodological)*, 23:414–422.

Crossbow, 2009. "Mica Motes," Technical report. Available at: http://www.xbow.com/Home/wHomePage.aspx.

Esram, T. and Chapman, P. 2007. "Comparison of Photovoltaic Array Maximum Power Point Tracking Techniques," *IEEE Transactions on Energy Conversion*, 22:439–449.

Farrar, C.R. and Worden, K. 2007. "An Introduction to Structural Health Monitoring," *Philosophical Transactions of the Royal Society*, 365:303–315.

FreeRTOS. 2009. "The freertos.org project," Technical report. Available at: http://www.freertos.org/.

HPWREN. 2009. "High Performance Wireless Research and Education Network," Technical report. Available at: http://hpwren.ucsd.edu/.

Hunter, J.S. 1986. "The Exponentially Weighted Moving Average," *Quality Technology*, 18:203–207.

Instruments, T. 2009. "Tms320c2811 Digital Signal Processor," Technical report. Available at: http://www.ti.com/.

Iqdour, R. and Zeroual, A. 2007. "Prediction of Daily Global Solar Radiation Using Fuzzy Systems," *International Journal of Sustainable Energy*, 26:19–29.

Kansal, A., Hsu, J., Zahedi, S. and Srivastava, M.B. 2007. "Power Management in Energy Harvesting Sensor Networks," *Transactions on Embedded Computing Systems*, 6:32.

Kansal, A., Potter, D. and Srivastava, M.B. 2004. "Performance Aware Tasking for Environmentally Powered Sensor Networks," In: *SIGMETRICS '04/Performance '04: Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems*, ACM, New York, NY, USA, pp. 223–234.

Leon-Garcia, A. 2008. *Probability, Statistics, and Random Processes for Electrical Engineering*, Pearson/Prentice Hall, Upper Saddle River.

Lin, K., Yu, J., Hsu, J., Zahedi, S., Lee, D., Friedman, J., Kansal, A., Raghunathan, V. and Srivastava, M. 2005. "Heliomote: Enabling Long-lived Sensor Networks Through Solar Energy Harvesting," In: *SenSys '05: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, ACM, New York, NY, USA, p. 309.

Magno, M., Brunelli, D., Zappi, P. and Benini, L. 2008. "A Solar-powered Video Sensor Node for Energy Efficient Multimodal Surveillance," In: *DSD '08: Proceedings of the 2008 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools*, IEEE Computer Society, Washington, DC, USA, pp. 512–519.

Marbach, P. 2007. "Distributed Scheduling and Active Queue Management in Wireless Networks," In: *Proceedings of 26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, Anchorage, Alaska, USA, pp. 2321–2325.

Mateu, L., Codrea, C., Lucas, N., Pollak, M. and Spies, P. 2007. "Human Body Energy Harvesting Thermogenerator for Sensing Applications," In: *SENSORCOMM '07: Proceedings of the 2007 International Conference on Sensor Technologies and Applications*, IEEE Computer Society, Washington, DC, USA, pp. 366–372.

Maxstream. 2009. "XBee OEM RF Module," Technical report. Available at: http://www.digi.com/products/wireless/point-multi-point/xbee-pro-series1-module.jsp.

Meninger, S., Mur-Miranda, J., Amirtharajah, R., Chandrakasan, A. and Lang, J. 2001. "Vibration-to-electric Energy Conversion," *IEEE Transactions on Very Large Scale (VLSI) Integration Systems*, 9:64–76.

Morais, R., Matos, S.G., Fernandes, M.A., Valente, A.L.G., Soares, S.F.S.P., Ferreira, P.J.S.G. and Reis, M.J.C.S. 2008. "Sun, Wind and Water Flow as Energy Supply for Small Stationary Data Acquisition Platforms," *Computers and Electronics in Agriculture*, 64:120–132.

Moser, C., Brunelli, D., Thiele, L. and Benini, L. 2006. "Lazy Scheduling for Energy-harvesting Sensor Nodes," In: *Fifth Working Conference on Distributed and Parallel Embedded Systems (DIPES'06)*, Braga, Portugal, pp. 125–134.

Musiani, D., Lin, K. and Rosing, T.S. 2007. ''Active Sensing Platform for Wireless Structural Health Monitoring,'' In: *Proceeding of 6th International Symposium on Information Processing in Sensor Networks IPSN 2007*, Cambridge (MIT Campus), Massachusetts, USA, pp. 390–399.

Park, C. and Chou, P. 2006. ''Ambimax: Autonomous Energy Harvesting Platform for Multisupply Wireless Sensor Nodes,'' In: *Proceedings of 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks SECON '06*, Reston, VA, USA, Vol. 1, pp. 168–177.

Park, C., Xie, Q., Chou, P.H. and Shinozuka, M. 2005. ''Duranode: Wireless Networked Sensor for Structural Health Monitoring,'' In: *Proceedings of IEEE Sensors*, Hyatt Regency Irvine, California, USA, p. 4.

Park, G., Sohn, H., Farrar, C.R. and Inman, D. 2003. ''Overview of Piezoelectric Impedance-based Health Monitoring and Path Forward,'' *The Shock and Vibration Digest*, 35:451–463.

Pon, R., Batalin, M.A., Gordon, J., Kansal, A., Liu, D., Rahimi, M., Shirachi, L., Yu, Y., Hansen, M., Kaiser, W.J., Srivastava, M., Sukhatme, G. and Estrin, D. 2005. ''Networked Infomechanical Systems: A Mobile Embedded Networked Sensor platform,'' In: *Proceedings of Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*, UCLA, Los Angeles, California, USA, pp. 376–381.

PowerFilm. 2009. ''OEM Components,'' Technical report. Available at: http://www.powerfilmsolar.com/.

Raghunathan, V. and Chou, P.H. 2006. ''Design and Power Management of Energy Harvesting Embedded Systems,'' In: *Proceedings of International Symposium on Low Power Electronics and Design (ISLPED 2006)*, Tegernsee, Germany, pp. 369–374.

Raghunathan, V., Kansal, A., Hsu, J., Friedman, J. and Srivastava, M. 2005. ''Design Considerations for Solar Energy Harvesting Wireless Embedded Systems,'' In: *Proceedings of Fourth International Symposium on Information Processing in Sensor Networks (IPSN 2005)*, 24–27 April, Los Angeles, California, pp. 457–462.

Raghunathan, V., Schurgers, C., Park, S. and Srivastava, M.B. 2002. ''Energy-aware Wireless Microsensor Networks,'' *IEEE Signal Processing Magazine*, 19:40–50.

Raghunathan, V., Spanos, P. and Srivastava, M. 2001. ''Adaptive Power-fidelity in Energy-aware Wireless Embedded Systems,'' In: *Proceedings of 22nd IEEE Real-time Systems Symposium (RTSS'01)*, London, UK, pp. 106–115.

Rong, P. and Pedram, M. 2003. ''Extending the Lifetime of a Network of Battery-powered Mobile Devices by Remote Processing: A Markovian Decision-based Approach,'' In: *Proceedings of Design Automation Conference*, Anaheim, CA, USA, pp. 906–911.

Savvides, A. and Srivastava, M.B. 2002. ''A Distributed Computation Platform for Wireless Embedded Sensing,'' In: *Proceeding of IEEE International Conference on Computer Design: VLSI in Computers and Processors*, Washington, DC, IEEE Computer Society, pp. 220–225.

Shuman, D. and Liu, M. 2006. ''Optimal Sleep Scheduling for a Wireless Sensor Network Node,'' In: *Proceedings of Fortieth Asilomar Conference on Signals, Systems and Computers ACSSC '06*, Pacific Grove, CA, pp. 1337–1341.

Sibley, G.T., Rahimi, M.H. and Sukhatme, G.S. 2002. ''Robomote: A Tiny Mobile Robot Platform for Large-scale Ad-hoc Sensor networks,'' In: *Proceedings of IEEE International Conference on Robotics and Automation ICRA'02*, 2, Washington D.C., USA, Vol. 2, pp. 1143–1148.

Simjee, F. and Chou, P.H. 2006. ''Everlast: Long-life, Supercapacitor-operated Wireless Sensor Node,'' In: *Proceedings of International Symposium on ISLPED'06 Low Power Electronics and Design*, Tegernsee, Bavaria, Germany, pp. 197–202.

Suehrcke, H. and McCormick, P. 1992. ''A Performance Prediction Method for Solar Energy Systems,'' *Solar energy*, 48:169–175.

Vigorito, C.M., Ganesan, D. and Barto, A.G. 2007. ''Adaptive Control of Duty Cycling in Energy-harvesting Wireless Sensor Networks,'' In: *Proceedings of 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks SECON '07*, San Diego, California, USA, pp. 21–30.

Yuan, L. and Qu, G. 2007. ''Alt-dvs: Dynamic Voltage Scaling with Awareness of Leakage and Temperature for Real-time Systems,'' In: *Proceedings of Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS'07)*, University of Edinburgh, Scotland, UK, pp. 660–670.